REMARKS/ARGUMENTS

The office action of May 13, 2004 has been carefully reviewed and these remarks are responsive thereto. Reconsideration and allowance of the instant application are respectfully requested. Claims 1-11 and 17-36 remain in this application. Claims 12-16 have been canceled without prejudice or disclaimer and new claims 37-39 have been added.

Fig. 1 has been objected to for failing to comply with margin requirements for formal drawings. Applicants are submitting herewith a replacement Fig. 1 to correct the margins. In addition, applicant has renumbered the labels associated with the IEEE 1394 Interface (formerly 140) and devices (formerly 150) as these labels were duplicative of ROM 140 and RAM 150. In addition, applicant has amended the specification to change the labeling as well. Also, other amendments have been made to the specification to correct other clerical errors.

Applicant has amended claims 19, 27 and 28 to improve clarity and not for reasons related to patentability.

The claims stand rejected under 35 U.S.C. § 103(a) as being unpatentable over the following combinations:

a. claims 1-3, 6-11, 17, 18 and 33-36 over U.S. patent no. 5,717,903 to Bonola in view of U.S. patent no. 6,202,155 to Tushie et al. ("Tushie") and further in view of U.S. patent no. 5,968,152 to Staats and further in view of "OFFICIAL NOTICE;"
b. claims 4 and 5 over the combinations of Bonola, Tushie and Staats and further in view of U.S. patent 5,996,050 to Carter et al. ("Carter");
c. claims 12, 13 and 15 over Tushie in view of U.S. patent 6,345,241 to Brice et al. ("Brice");
d. claims 14 and 16 over the combination of Tushie and Brice and further in view of U.S. patent no. 6,327,637 to Chang;
e. claims 19-21, 26, 27, 29 and 32 over Staats in view of Bonola;
f. claims 22 and 23 over the combination of Staats, Bonola and Carter;
g. claims 24 and 25 over the combination of Staats, Bonola, Carter and Tushie;
h. claim 28 over the combination of Staats, Bonola and Carter; and

i. claims 30 and 31 over the combination of Staats, Bonola and Tushie.

Applicant respectfully traverses these rejections.

Claims 1-11, 17, 18 and 33-36

Regarding independent claim 1, the action alleges that Bonola, at col. 1, line 65 to col. 2, line 32, discloses a method of emulating a device including loading an emulation driver for the device and dynamically exposing the emulated device functionality. The action acknowledges however that Bonola fails to teach or suggest creating a virtual device object for the device and emulating a device by a node on a serial bus. To show creating a virtual device object for the device, the action relies on Tushie at col. 3, lines 19-24 and col. 4, lines 45-55. To show emulating a device by a node on a serial bus, the action relies on Staats claiming it discloses device drivers providing support for devices on nodes of a serial bus pointing to figures 1, 4, 7 and col. 2, lines 44-52.

Bonola discloses emulating a peripheral device to allow device driver development before availability of the peripheral device. Bonola notes that often the peripheral device may be unavailable while the device is being developed. By emulating the peripheral device, testing and debugging of the device driver can be performed including testing of features such as multi-tasking, multi-threading and real time operations. According to Bonola, the virtual device emulator (VDE) is loaded by a host microprocessor into system memory. Col. 2, lines 4-6; col. 6, lines 18-20. In a system with four microprocessors, the VDE can be divided into different tasks and executed on up to three free target microprocessors that are not being accessed by the operating system. Col. 2, lines 8-12; col. 6, lines 5-11. The VDE includes an emulator loader which is run on the host CPU and operating system dependent, and which loads the emulator into system memory for execution by the target CPUs. Col. 2, lines 17-20. In addition to the VDE, the Bonola system includes separate device drivers, which execute two sets of macro commands, one set if the emulation mode is selected and another set if emulation mode is not selected. In contrast to the action's assertion, Bonola does not teach or suggest loading an emulation driver for the device. Bonola merely describes loading emulation software for the peripheral device. Neither Tushie nor Staats overcomes this deficiency. For at least this reason, the combination of

Bonola, Tushie and Staats, even if proper, does not result in the invention recited in independent claims 1 and 33.

In addition, Bonola does not teach or suggest a method of implementing an emulation driver as recited in independent claim 17. As discussed above, Bonola merely describes loading emulation software for the peripheral device and provides separate device drivers, which execute two sets of macro commands, one set if the emulation mode is selected and another set if emulation mode is not selected. It follow that Bonola lacks a teaching or suggestion of allocating node address space to intercept requests to an emulated device register. Indeed, the action does not point with specificity to any portion of Bonola, Tushie or Staats for this claim 17 feature.

The action alleges that one skilled in the art with the disclosure of Bonola would have been motivated to look for "object oriented methods of abstracting computer peripheral devices in software for the purpose of developing device drivers" before the peripheral devices were available." *Action*, p. 3. The action further avers that one skilled in the art would have combined device emulation of Bonola with the object oriented abstraction of Tushie "because, by allowing for easy configuration change to take place on a virtual model of a device rather than the device itself, changes can be made without having to use a complex programming language which would add complexity and expense." *Id.*

Contrary to the action's position, Applicant respectfully submits that one skilled in the art would not have been motivated to combine Bonola and Tushie. The action contends that one would have used the virtual device object described in Tushie with the emulation technique of Bonola so that a configuration change can take place on the device model rather than the device itself. The problem with this position is that one of the primary reasons that Bonola emulates the peripheral device is to allow for device drivers to be tested without the device itself. Indeed the emulation software of Bonola could be easily modified without having to use a complex programming language which would add complexity and expense, and without the need to utilize virtual device objects of Tushie as proposed in the action. In short the motivation for combining the Bonola and Tushie proposed in the action is illusory. Staats fails to mitigate this defect with the propriety of the combination. For at least this reason, one would not have combined Bonola, Tushie and Staats to obtain the invention of independent claims 1, 17 and 33.

In view of the above reasons, independent claims 1, 17 and 33 are patentably distinguishable over the combination of Bonola, Tushie and Staats. The Official Notice of the IEEE 1394 Bus Standard referred to in the action was only specifically applied to reject independent claim 17. In any event, it does not remedy any of the deficiencies discussed above. Claims 2, 3, and 6-11, which directly or indirectly depend on claim 1, claim 18, which depends on claim 17, and claims 34-36, which directly on indirectly depend on claim 33, are considered allowable for the same reason as their ultimate base claim and further in view of the additional features recited therein. For example, claim 3 calls for creating, by the at least one other node, a physical device object for the device, and loading a device driver for the device. The action points to Fig. 2, item 203 of Tushie to show a physical device object. By its very name, item 203 involves a virtual card object rather than a physical device object.

To reject claims 4 and 5, which ultimately depend from claim 1, the action combines Carter with Bonola, Tushie and Staats. However, Carter fails to overcome any of the deficiencies noted above with respect to Bonola, Tushie and Staats. For at least this reason, the combination of Bonola, Tushie, Staats and Carter does not result in the invention of claims 4 and 5.

Claims 12-16

Applicant has canceled claims 12-16 without prejudice or disclaimer thereby rendering the rejection of these claims moot.

Claims 19-32

Independent claim 19 is directed to a system for emulating a device and calls for a serial bus and a node connected to the serial bus which is configured to emulate at least one device. The action contends and applicant agrees that Staats discloses all the claim 19 features, but for the node emulating a device. To overcome this deficiency, the action relies on Bonola. Bonola describes emulating a peripheral device to allow device driver development before availability of the peripheral device.

To justify the combination, the action avers that Staats identifies that problems exist in the addition of future devices due to limited key space and that a driver image can be loaded from media that is plugged into the bus. The action goes on to allege:

> An artisan of ordinary skill, knowing the *IEEE 1394* standard bus architecture accommodates hot removal and insertion of devices would want a method of

> providing a "*place holder*" for a device that has been temporarily removed but which can reasonably be assumed will be re-inserted at a future time. The artisan would not want the user of the computer system to have to wait while the device drive for the device at the particular node would have to be loaded and unloaded every time that particular peripheral was inserted into the bus. Thus, an artisan would look for a mechanism to represent that device on the serial bus with out the actual device actually being present. In the related art of peripheral emulation, the *Bonola* reference teaches the emulation of a peripheral device that is not actually present. [Citation Omitted].
>
> Thus, it would have been obvious, to one of ordinary skill in the art, at the time the invention was made, to have combined the serial bus methods of the *Staats* reference with the peripheral emulation method of the *Bonola* reference because, by emulating a peripheral on the 1394 bus, the device driver for that device doesn't have to be reloaded every time the device is inserted into the bus, thus allowing the computer system to operate more efficiently.

Office Action, paper no. 7, p. 10. Thus, the action has identified an alleged motivation for combining the references based on the common knowledge of one of ordinary skill in the art. Applicant respectfully disagrees with the action's analysis that one skilled in the art would have motivated to modify Staats with Bonola as asserted in the action. In this regard, Staats discloses a methodology of creating true "plug and play" capabilities using extended key values within the configuration ROM hierarchical structure. Staats, at col. 6, lines 16-30 provides that:

> One example of the use of extended key values concerns directory structures for device drivers. For true "plug and play" operation, each device (or unit) of computer system 50 must be able to provide its own device driver so that an operating system can load and use that driver without the need for manual driver installation. Drivers for multiple operating systems may be provided by a device. Information about these drivers can be stored in a directory in a CSR ROM. The actual drivers may be stored in the CSR ROM, from which they can be loaded directly, or the drivers could be made available over the internet, referenced by a URL, or by some other loading mechanism. To accommodate these multiple drivers, a device driver directory may be established within each unit directory in a CSR ROM so that nodes with multiple units can make multiple drivers available.

As described in Staats, information for each device driver can be stored in the CSR ROM. Thus, if a device were unplugged (hot removal) from the serial bus, it could be inserted back into the bus for true "plug and play" operation. The action has alleged that the user of the computer

system would not want to wait while the device driver for a device is loaded every time that device was inserted into the bus. First there is no suggestion in any of the references that a problem exists in that a user has to wait while a device driver is loaded after the corresponding device is inserted into the bus and that doing so would be efficient as alleged in the action. Quite the contrary, Staats has provided a system where different device drivers can be loaded upon insertion of devices to provide true "plug and play", namely where "plug and play" operation occurs in real time. In sum, applicant believes that Staats provides sufficient information to teach away from the motivation identified in the action. Indeed, it would appear that the action engaged in an exercise of impermissible hindsight. Namely, with the claim 19 invention in hand and the references, the action tried to create a reason to combine Staats and Bonola. In any event, applicants submit that one skilled in the art would not have had motivation to combine Staats and Bonola as asserted in the action to obtain the claim 19 invention.

Claim 27 is directed to a device configured to emulate at least one other device. The action asserts that Staats discloses all the features of claim 27, but for emulation of a device. As with claim 19, to overcome this deficiency, the action relies on Bonola. As discussed with respect to claim 19, one would not have been motivated to modify Staats with Bonola to obtain the claim 27 invention.

Claims 20-26, which depend from claim 19, and claims 28-32, which depend from claim 27, are patentable over the applied art for at least the same reasons as their base claim and further in view of the novel and non-obvious features recited therein. Carter and Tushie fail to overcome the deficiencies noted with respect to the combination of Staats and Bonola. For example, claim 25 calls for the device driver to be a virtual device driver. The action relies on Fig. 2 and col. 3, lin3w 10-24 of Tushie to show this feature, yet the cited portion of Tushie merely describes virtual device objects and not virtual device drivers as recited in claim 25.

New Claims 37-39

New claims 37-39 are fully supported by the specification and are patentably distinct over the art of record for the same reasons as their base claim and further in view of the features recited therein.

Claim 37 calls for, among other features, creating a virtual device object for the device occurs without the device being connected to the node. The action relies on Tushie to show creating a virtual device object. Admittedly, Tushie contemplates virtual device objects. Tushie discloses a virtual card personalization system, which receives information to personalize a transaction card and creates a virtual transaction card using the information. The portion relied on by the action in relevant part states:

> [T]he user manipulates the virtual card object to define the desired transaction card and configures the virtual device objects to *reflect the physical personalization equipment present in the production environment* through interactive input or simple scripts. When the production environment changes, the user simply changes the configuration of the corresponding virtual device object(s). Thus, the virtual card personalization system presents a conceptual model of the personalization process to the user and permits the user to quickly and easily change the model without having to resort to complicated programming languages or understand complex file formats. (Emphasis added)

Tushie, col. 3, lines 12-23. The virtual card personalization system includes a virtual card object and virtual personalization machine (device) object and controls physical devices, such as card printers, embossing devices, ATMs and PCs among others, referred to as personalization equipment. Referring to figure 4, the virtual personalization machine object 205 includes virtual device sub-objects 401-407. The virtual device sub-objects are used to control corresponding physical devices which comprise the personalization equipment 130. Col. 6, lines 45-48.

In contrast to the claim 37 invention however, Tushie describes in the cited portion set for the above that the virtual device objects are configured to reflect the physical personalization equipment *present* in the production environment. Tushie further states that "[w]hen the production environment changes, the user simply changes the configuration of the corresponding virtual device object(s)." Col. 3, lines 16-18. It is quite clear that Tushie neither teaches nor suggests creating a virtual device object for the device without the device being connected as recited in claim 37.

CONCLUSION

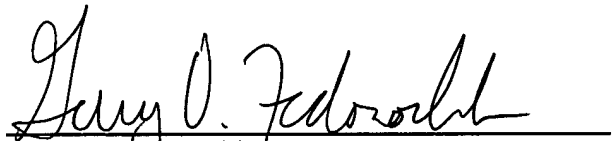
It is believed that no fee is required for this submission. If any fees are required or if an overpayment is made, the Commissioner is authorized to debit or credit our Deposit Account No. 19-0733, accordingly.

All rejections having been addressed, applicant respectfully submits that the instant application is in condition for allowance, and respectfully solicits prompt notification of the same.

Respectfully submitted,

BANNER & WITCOFF, LTD.

Dated: August 13, 2004

By: 

Gary D. Fedorochko
Registration No. 35,509

1001 G Street, N.W.
Washington, D.C. 20001-4597
Tel: (202) 824-3000
Fax: (202) 824-3001

GDF:lab

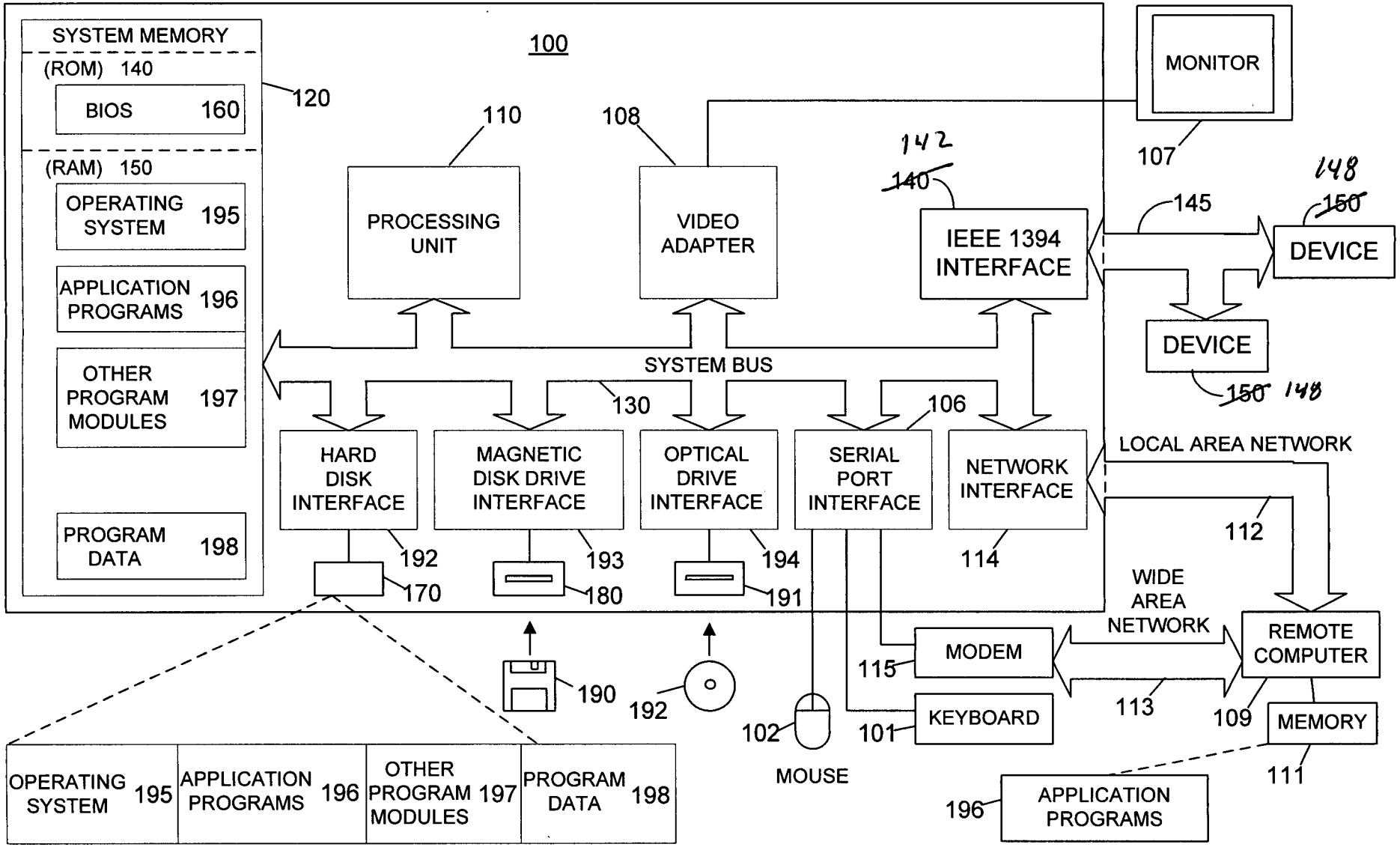


FIG. 1